MEMO #94-02
January 1994

A System for the Off-Line Recognition of Handwritten Text

Thomas M. Breuel

IDIAP, C.P. 609, 1920 Martigny Switzerland
Tel: +41 (26) 22 76 64, FAX: +41 (26) 22 78 18
tmb@maya.idiap.ch

ABSTRACT

A new system for the recognition of handwritten text is described. The system goes from raw, binary scanned images of census forms to ASCII transcriptions of the fields contained within the forms. The first step is to locate and extract the handwritten input from the forms. Then, a large number of character subimages are extracted and individually classified using a MLP (Multi-Layer Perceptron). A Viterbi-like algorithm is used to assemble the individual classified character subimages into optimal interpretations of an input string, taking into account both the quality of the overall segmentation and the degree to which each character subimage of the segmentation matches a character model. The system uses two different statistical language models, one based on a phrase dictionary and the other based on a simple word grammar. Hypotheses from recognition based on each language model are integrated using a decision tree classifier. Results from the application of the system to the recognition of handwritten responses on U.S. census forms are reported.

1 Introduction

The recognition of handwritten text is an important practical problem. Its applications include bank checks, postal mail addresses, census and poll

A version of this paper has been submitted to ICPR'94.

forms, tax forms, office documents, library information systems, FAX routing, personal organizers, and many more.

It is also an interesting problem in computational vision, since it encompasses many of the major problems in computational vision: feature extraction, figure-ground problems, segmentation, learning, idiosyncratic and "natural" shape variation, and the integration of top-down knowledge. From an experimental point of view, the recognition of handwritten text is a particularly nice model system, since it is a well-defined problem and large amounts of training and test data are available.

This paper describes a new system for the recognition of handwritten text. The system is a complete forms-to-ASCII system. Information flow in the system is strictly bottom-up; that is, each stage computes a complete representation of all the information necessary for subsequent processing stages.

Four major processing stages can be distinguished. *Preprocessing* starts with the raw input and computes images of isolated, normalized strings of handwritten text found within the raw input. The *segmentation* stage divides up the handwritten text into potential character subimages and describes the spatial relations among those character subimages compactly using a graph structure. The *recognition* stage determines how similar each character subimage is to known, well-segmented character subimages in the database. Finally, in the *postprocessing* stage, the individual character subimages are assembled into a globally optimal interpretation of the handwritten input string, taking into account constraints imposed by the language model. Intermediate results from the various processing stages are shown in Figure 2.

2 Statistical Foundations

The statistical basis for the system described in this paper is very similar to that of segment-based speech recognition systems. The presentation and notation used below follow the paper by Leung *et al.*, 1991, very closely.

The goal of the system is to identify the character string $W = w_1 \dots w_n$ that represents the most probable interpretation of the input image x. More precisely, in the set of all permissible strings Σ^*, we want to find the string $W \in \Sigma^*$ that maximizes $P(W|x)$. In a Bayesian framework, this represents the optimal decision under a zero-one loss function.

We can think of this interpretation of the input image as a character string as consisting of two parts: a segmentation $S = s_1 \dots s_n$ of the input image, that is, a partitioning of the image into character subimages, and an interpretation of each character subimage s_i as a character w_i.

Consider now the joint conditional distribution $P(W,S|x)$. The desired probability $P(W|x)$ is related to this by summing over all possible segmentations:

$$P(W|x) = \sum_S P(W,S|x) \tag{1}$$

We impose the *a priori* constraint that $P(W,S|x) = 0$ if $|W| \neq |S|$. That is, the number of character subimages generated by the segmentation and the length of the hypothesized string must be the same.

In our system, we will be using context-independent models for the character subimages. Therefore, we want to express $P(W,S|x)$ in terms of the individual w_i and s_i:

$$\begin{aligned}
P(W,S|x) &= \frac{P(x|W,S)\,P(W,S)}{P(x)} && (2)\\
&= \frac{P(x|w_1,\ldots,w_n,s_1,\ldots,s_n)\,P(s_1,\ldots,s_n|w_1,\ldots,w_n)\,P(W)}{P(x)} && (3)\\
&\approx P(W)\prod_i \frac{P(x_i|w_i,s_i)\,P(s_i|w_i)}{P(x_i)} && (4)\\
&= P(W)\prod_i \frac{P(w_i,s_i|x_i)}{P(w_i)} && (5)
\end{aligned}$$

Here, the approximation is based on the assumption of context independence. The variables x_i refer to those parts of the input vector x that correspond to segment s_i.

Leung *et al.*, 1991, go on to rewrite the probability $P(w_i,s_i|x_i)$ to include contextual dependencies and to separate the segment classification and segmentation contributions to the overall probability.

In this work, we take a simpler approach and estimate $P(w_i,s_i|x_i)$ directly. In particular, we do not attempt to compute probabilities for the placing of segmentation boundaries or cuts at particular points, but instead we estimate the probability that each of the resulting character subimages is part of a correct segmentation.

To illustrate this, let us assume that we are just modeling the probability $P(S|x)$ of the segmentation S given the data x. Then, making the same independence assumption as above, we can write

$$P(S|x) \approx \prod_i P(s_i|x) \tag{6}$$

where each s_i corresponds to a character subimage. This clearly represents only an approximation, since moving the boundary between character subimage s_i and s_{i+1} is almost certainly going to affect both $P(s_i|x)$ and $P(s_{i+1}|x)$ in a correlated way. However, this approximation appears to be sufficiently good, since the major function of the contribution of $P(S|x)$ to the overall recognition problem is to strongly penalize segmentations containing character subimages representing more than a single character; this can be accomplished well even under the independence assumption

3 Database and Input Data

The database used in this work was made available by NIST for participation in the Second Census OCR Conference (R. Allen Wilkinson *et al.*, 1994). In the work described here, a subset of 1200 form images digitized from paper is used for training and testing. These form images can be found in the subdirectories of the `data2` directory of the CD-ROM containing the "NIST Special Database 12/Miniform Database 2 of Binary Images from Microfilm and Paper". This set of 1200 form images will henceforth be referred to simply as "the NIST database".

Each form image consists of 5 miniforms. An example of a miniform is shown in Figure 1. Each miniform contains the responses to three questions about the nature of the employment of the respondent. Therefore, a total of 18000 handwritten strings are available for training and testing.

The forms in the database represent actual responses received by the census bureau. Hence, there are no constraints on the vocabulary, grammar, spelling, writing instruments, or writing styles.

For each field, the database contains a transcription of the alphanumeric characters contained in that field. Non-alphanumeric characters (e.g., "/" and "&") are not transcribed. Transcriptions contain spaces, but these often fail to represent physical spaces present in the input and sometimes transcribe space that is not actually present in the input. These transcriptions were used for training and testing.

4 Form Segmentation

Before any character recognition can take place, handwritten character strings need to be extracted from the miniforms. The first step of this extraction process is to locate approximately the boxes containing the handwritten responses.

This is done by first locating the prominent black lines that form part of the layout using morphological methods. The input boxes are then located relative to these lines. Form segmentation is complicated by the fact that the database contains two different layouts for miniforms, and that spurious lines appear on occasion due to noise and poor image quality. To cope with both problems, a 1D analog of a search-based object recognition algorithm (Grimson and Lozano-Perez, 1987) is used.

Out of a sample of 3000 miniforms, 2 miniforms failed to get extracted. These cases were detected automatically, and the corresponding fields were rejected.

In addition, each input field is surrounded by a printed, dashed box. Frequently, this box touches or crosses the handwritten input. The current system uses a morphological method for removing the dashed box while preserving strokes in the handwritten input. Essentially, the dashed box is first located by morphological filtering that selects the individual dashes. Then, the dashes forming the top and bottom of the printed input box are removed by removing any stretch of pixels that is part of the printed input box and has a vertical extent smaller than some empirically chosen threshold.

5 Skew and Slant Correction

Skew is the misalignment of the baseline of the handwritten text with respect to the pixel coordinate system of the field image. Slant is the deviation from the vertical of the long down strokes in letters like "l", "K", or "T". Skew correction shears the input image in a direction parallel to the y axis to force the skew of the input to zero. Slant correction shears the input image in a direction parallel to the x axis to force the average slant of the handwritten input to zero.

Skew and slant correction are each carried out by a generate-and-test procedure. We can think of this as generating different instances of the image subjected to skewing or slanting, and choosing the transformation that maximizes an evaluation function.

The evaluation function used in this work consists of the average value of the local maxima in the smoothed horizontal (skew correction) or vertical (slant correction) histogram.

The purpose of skew correction in the current system is to facilitate the identification of ascenders and descenders, one of the inputs to the character classifier. The purpose of slant correction is to allow the system to rely on vertical cuts to separate individual characters in the handwritten input; if slant were not corrected, cuts would have to be slanted.

6 Segmentation

The output of the previous stage is an image that primarily contains a handwritten input string. In addition, it may contain a few clusters of pixels representing noise, parts of the input box that failed to be removed, or part of a printed arrow protruding into the input box. The goal of the segmentation stage is to determine a collection of character subimages that can then be individually classified.

The first part of the segmentation step is to find all connected components in the input image. Unfortunately, it is exceptional that the connected components correspond directly to characters. In many cases, even in "isolated" handprinted character writing styles characters are connected through ink trails or noise. Conversely, characters like "E", "T", or "H" often consist of two or more connected components. Because of pen fading, in particular during long, smoothly curved strokes, characters like "N" and "U" also can appear separately, i.e., as two or more connected components, in the image.

Connected components are therefore subjected to a grouping process. We define a *central region* as those parts of the input image that lie between the 20th and 80th percentile of the horizontally projected histogram. Connected components in the input image that overlap the central region are automatically retained. Other connected components are retained only if their bounding boxes overlap significantly one of the connected components overlapping the central region. Connected components that are not retained during these two steps are deleted and not considered further by subsequent processing stages.

The connected components analysis also finds potential spaces in the input. The locations of spaces in the input are recorded; they later participate in matching and recognition.

Since individual connected components may represent several touching input characters, they need to be subdivided further. This is achieved by cutting apart the handwritten input just to the left of every vertical stroke. This approach works very well for handprinted upper case characters that have a vertical stroke (possibly curved or slanted) on the left (like "A", "B", or "C"). It works acceptably well with the remaining upper case characters and most lower case characters. The only significant exception is the letter "T", which, when cut right before its vertical stroke, will leave a horizontal line attached to the preceding letter.

Strokes are identified as local maxima in a vertical histogram, and stroke boundaries are found as inflection points, i.e., zeros of the second derivative, of the smoothed vertical histogram. Potential vertical cutting lines through the handwritten input are defined at each stroke boundary to the left of a stroke that has a vertical projection consisting of more pixels than an empirically chosen threshold. To simplify the handling of boundary conditions, two extra cuts are created bracketing the complete handwritten input.

The pixels between any pair of vertical cutting lines (not necessarily adjacent) form a character subimage. All character subimages satisfying an empirically chosen size constraint are retained. In particular, a character subimage must not be wider than three times the estimated text height.

Clearly, using only vertical cuts would mean that the system would have difficulties with kerned letter pairs, like "To". In the system, cuts are therefore modified to separate most kerned letter pairs cleanly. Due to lack of space, the method used for doing this will be described elsewhere.

The set of all character subimages and their spatial relationships is represented as a directed graph, the hypothesis graph. Each node in the hypothesis graph corresponds to a vertical cut. An edge from s to t corresponds to the character subimage that is delimited on the left by cut s and on the right by cut t. A segmentation of the input image into individual characters now corresponds to a simple path through the hypothesis subgraph.

7 Character Subimage Classification

Each character subimage found by the segmentation stage is individually classified using a MLP (Multi-Layer Perceptron with sigmoidal activation functions). As is well-known, the output of the classifier approximates a conditional probability $P(w_i|x)$ (see Bourlard and Wellekens, 1989, Bridle, 1990, Bourlard *et al.*, 1992, and Renals and Morgan, 1992, for similar applications and further references).

The possible classes $w_i \in \Sigma$ are the 26 letters "A" through "Z" (no distinction is made between upper and lower case) and a rejection class. The rejection class contains all non-alphabetic characters and all character subimages that do not represent a complete, single character. For the recognition of U.S. census forms, the system was not trained to recognize digits or special characters: the number of digits in the training set was too low to warrant inclusion, and special characters were not transcribed, making training difficult.

Ideally, only character subimages belonging to the correct segmentation would be classified as one of the letters "A" through "Z"; all character subimages belonging to incorrect segmentations should be assigned to the rejection class by the classifier. Of course, as we observed above, this ideal is not achievable, since the same string frequently allows several plausible segmentations. Such ambiguities must be resolved on the basis of top-down knowledge.

Training of the MLP was carried out using the backpropagation algorithm with a momentum term (Rumelhart *et al.*, 1986, Hertz *et al.*, 1991). Training was stopped when the cross-validated error did not decrease further.

Input to the classification stage consists of eight normalized feature maps, the first seven of which are 10×10 unit topographic representations of feature maps corresponding to the character subimage. The eighth feature map encodes the ascent, descent, width, height, and center relative to the baseline of the character subimage using a unary code.

The first four feature maps encode the local gradient of the character subimage. Each feature map is maximally sensitive to a particular gradient orientation; the response to gradients differing from this preferred orientation decays like a Gaussian.

The next feature map encodes the presence of "holes", i.e., interior regions that are not connected to the background of the character subimage. Such regions occur frequently in letters like "O", "a", "A", or "e", and are almost always absent in letters like "L", "T", or "I".

The last two feature maps encode the presence of singular points of the skeleton of the character subimage. The first of the two feature maps encodes endpoints of the skeleton, while the second encodes points where three or four branches of the skeleton meet. The skeleton is computed using the thinning algorithm described in Pavlidis, 1982.

8 Hypothesis Matching

The output of the segmentation stage is a collection of character subimages, potential spaces and their locations, and an associated directed graph. The character subimage classification stage has associated a cost vector with each edge in this graph, which can be thought of as describing the cost of interpreting the corresponding character subimage as a character of class i. We note that we can think of this graph equivalently as a Hidden Markov Model (HMM) or a Finite State Machine (FSM).

The problem of interpreting the input string consists of picking a path through the hypothesis graph that starts at the vertex corresponding to the leftmost cut of the image of the handwritten input string and finishes at the vertex corresponding to the rightmost cut of the image of the handwritten input string. Each edge in such a path corresponds to the interpretation of a character subimage.

In practice, the hypothesis graph derived from the segmentation needs to be edited slightly to account for the possible insertion and deletion of character subimages from the input string. For example, a non-negligible fraction of the input images contain spurious blobs before and after the actual handwritten string. In addition, some input strings contain insertions of special characters like "/", "&", and "-". The unedited hypothesis graph above would force the interpretation of such extraneous character subimages as characters or as parts of other character subimages, resulting in incorrect hypotheses.

If we view the hypothesis graph as a representation of a FSM or HMM, we can overcome this problem by adding ϵ-transitions and self-loops to each state. In many HMM-based speech recognition systems, the estimation of the probabilities of such transition is very important, since they represent durational models for phonemes in the input. In this system, costs associated with ϵ-transitions and self-loops were picked on the basis of some simple experimentation. The reason why this seems to be sufficient is that ϵ-transitions and self-loops need to participate only rarely in a match, and "durational" models (i.e., character width models) are already implicit in the segmentation and character subimage classification steps.

9 Dictionaries

In principle, we could simply pick the best path through the hypothesis graph and use that as our transcription of the input string. Unfortunately, recognition performance using such an approach is generally poor. The reason is that most handwritten input is ambiguous. These ambiguities can only be resolved using a language model that restricts the set of paths through the hypothesis graph to those that are compatible with strings given by the language model.

Manual transcriptions and dictionaries from 1980 and 1990 census forms were supplied with the NIST database. From this data, two language models were constructed. The first consists of approximately 20000 complete phrases and estimated associated frequency information and has a coverage of 66% of the input phrases encountered by the system. The second consists of unconstrained concatenations of 15000 frequently used words, separated by spaces, and uses a word insertion penalty to assign probabilities to phrases. It has higher coverage (87%) than the phrase-based language model but contains many implausible or impossible phrases.

An optimal interpretation of the hypothesis graph constrained by a language model is found using a Viterbi-style algorithm.

The system finds two hypotheses, one using the phrase-based language model and the other using the word-based language model. A decision between the two hypotheses is made using a decision tree. The decision tree itself has been obtained using the CART method (Breiman et al., 1984).

10 Bootstrapping and Training

The MLP classifier that performs character subimage classification requires training data. Except for the decision tree classifier, all the other parameters of the system have been set either by estimating probabilities using counting (e.g., phrase priors and character priors), by simple statistics on geometric measurements of the input, or by inspection.

Since the NIST database contains only transcriptions but no alignment information, training had to proceed in two phases. In the first phase, 1000 manually segmented and aligned input fields were used to train a crude MLP character subimage classifier. This initial classifier was then used to segment and align input strings, and character subimages from the resulting alignment were used to train better classifiers for character subimages in a process similar to embedded training in speech recognition. The MLP was trained on approximately 117000 character subimages in this way.

11 Results and Discussion

Subdirectory `d11` of the NIST database was never used for any training step, and was reserved instead for evaluating the performance of the system. All error rates and results presented below refer to the 1500 field images in this subdirectory.

In terms of throughput, the system is a research system and has not been optimized. Many computations are duplicated in different processing modules, and there is significant file I/O overhead. Keeping these caveats in mind, on a low-end SPARCstation (a SPARCstation ELC), form segmentation and pre-processing take less than 20 seconds per field, and segmentation and recognition take an average of 60 seconds per field and language model.

The system was evaluated based on its field error rate. A field was counted as classified correctly if the system returned a string that was an exact match against the string given in the transcription supplied with the NIST database, allowing for the insertion or deletion of spaces (note that the NIST criterion requires exact matches even for spaces). This means, in particular, that any misspelling in the handwritten input must be recognized and transcribed by the system.

Because the available language models have limited coverage, a certain error rate is intrinsic. For the phrase-based language model, the intrinsic error rate is 34%, and for the word-based language model, the intrinsic error rate is 13%. In addition, about 12% of the inputs are very difficult to recognize automatically because they are faded, written in a non-standard style, or contain manual corrections. We should therefore allow the system to reject unrecognizable or unknown input and evaluate its performance at various field rejection rates.

Error rates for the complete system and the phrase-based and word-based subsystems at various rejection rates are shown in Table 1. To get a better idea of which errors are due to an incomplete language model and which errors are due to other causes, Table 2 shows the error rates of each recognizer applied only to strings contained in its language model (but faded or otherwise difficult inputs are still present).

Comparing the performance of this system with that of other handwritten language recognition systems is difficult. Most importantly, other tasks usually involve small, fixed vocabularies. For example, in postal applications, closed dictionaries of size 10, 100, and 1000 are used for benchmarks, while financial applications often use simple grammars for written numbers based on less than 30 words.

For postal applications, Kimura *et al.*, 1993, report a zero rejection field error rate of 8.52% (down from 19.1% after extensive tuning) on the CEDAR database and using a dictionary of size 1000. Giloux *et al.*, 1993, report word-recognition error rates (a less stringent criterion) of 21% for a bank amounts recognition task, and 22.4% on a 100 city address recognition task.

The field error rates of 2.6% at 25% rejection and 12.4% at 0% rejection for the phrase-based recognizer applied to phrases in its language model are perhaps what is most comparable to those situtations. In comparing this with the postal applications, it should be kept in mind that this is for a dictionary about 20 times as large and for a new system that has not been tuned.

The field error rate at 50% rejection is one of the benchmarks used in the NIST conference and may represent the point at which commercial use of handwritten language recognition technology becomes economically interesting. The error rate of the current system is 6.1% at this rejection rate. The sources of errors for the current system were analyzed. It was found that nearly one third of the errors (2.0%) could probably have been avoided by better estimates of the per-character posterior probabilities. About one quarter (1.6%) are due to inputs not represented by either language model. Because the system must have high confidence in the resulting hypotheses, these errors tend to represent subtle emendations, like returning the hypothesis "ADMINISTRATION" for the misspelled input "ADMINSTRATION". About another quarter of the error rate (1.3%) is due to errors in pre-processing that resulted in the presentation in a partial or truncated image of the handwritten input to the recognizer. Such errors are either due to truncating the input on the right when writing falls outside the box, or removing one of the two lines of a two-line input field. Many of these errors are due to the particular layout of the input form and could be avoided by additional task-specific pre-processing and rejection of inputs.

Burges *et al.*, 1993, have independently developed a system that is very similar to the system described in this paper. They, too, perform character subimage classification and choose a globally optimal interpretation of the costs derived from the neural network output. They use a different segmentation method, though, and justify the statistical basis of their method only informally. On a closed lexicon of size 1000 for a U.S. postal address application, they achieve a recognition rate of 47%. The system described by Kimura *et al.*, 1993, is also similar to the system described in this paper. The main differences are in the way the handwritten input is segmented, the way individual character subimages are classified, and the way alternative segmentations are represented and evaluated. Several other systems for the recognition of connected handwritten input have recently been described (Edelman *et al.*, 1990, Guillevic and Suen, 1993, Caesar *et al.*, 1993, Keeler and Rumelhart, 1992, Martin and Rashid, 1992, Matan *et al.*, 1992). Another related system is that described in Schenkel *et al.*, 1993, which is applied to the on-line recognition of handwritten text. Each of them differs in important respects from the current system in terms of how the input is segmented, how character hypotheses are classified, and how character hypotheses are integrated with a language model.

In summary, the author believes that the excellent performance of this system on a real-world recognition task validates the approach. A more thorough evaluation of the performance of the system will be presented as part of the NIST conference (R. Allen Wilkinson *et al.*, 1994). Several of the techniques introduced by this system should prove useful for other recognition tasks involving handwritten, printed, or spoken language.

References

Bourlard H., Wellekens C. J., 1989, Links between Markov models and multilayer perceptrons, In Touretzky D. S., ed., *Advances in Neural Information Processing Systems*, volume 1, pages 502–510, San Mateo, CA, Morgan Kaufmann.

Bourlard H., Morgan N., Renals S., 1992, Neural nets and hidden markov models: Review and generalizations, *Speech Communication*, 11:237–246.

Breiman et al. L., 1984, *Classification and Regression Trees*, The Wadsworth statistics/probability series, Wadsworth, Belmont, CA.

Bridle J. S., 1990, Training stochastic model algorithms as networks can lead to maximum mutual information estimation of parameters., In Touretzky D. S., ed., *Advances in Neural Information Processing Systems*, volume 2, pages 211–271, San Mateo, CA, Morgan Kaufmann.

Burges C. J. C., Ben J. I., Denker J. S., LeCun Y., Nohl C. R., 1993, Off line recognition of handwritten postal words using neural networks, *International Journal of Pattern Recognition and Artificial Intelligence*, 7(4).

Caesar T., Gloger J. M., Mandler E., 1993, Preprocessing and feature extraction for a handwriting recognition system, In *International Conference on Document Analysis and Recognition*, pages 408–411, IEEE Computer Society Press.

Edelman S., Ullman S., Flash T., 1990, Reading cursive handwriting by alignment of letter prototypes, *International Journal of Computer Vision*, 5:303–331.

Giloux M., Leroux M., Bertille J.-M., 1993, Strategies for handwritten word recognition using hidden markov models, In *International Conference on Document Analysis and Recognition*, pages 299–304, IEEE Computer Society Press.

Grimson W. E. L., Lozano-Perez T., 1987, Localizing Overlapping Parts by Searching the Interpretation Tree, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(4):469–482.

Guillevic D., Suen C. Y., 1993, Cursive script recognition: A fast reader scheme, In *International Conference on Document Analysis and Recognition*, pages 311–314, IEEE Computer Society Press.

Hertz J., Krogh A., Palmer R. G., 1991, *Introduction to the Theory of Neural Computation*, Addison Wesley.

Keeler J., Rumelhart D. E., 1992, A self-organizing integrated segmentation and recognition neural net, In Moody J. E., Hanson S. J., Lippmann R. P., ed.s, *Advances in Neural Information Processing Systems*, volume 4, Morgan Kaufmann.

Kimura F., Shridhar M., Chen Z., 1993, Improvements of a lexicon directed algorithm for recognition of unconstrained handwritten words, In *International Conference on Document Analysis and Recognition*, pages 18–22, IEEE Computer Society Press.

Leung H. C., Hetherington I. L., Zue V. W., 1991, Speech recognition using stochastic explicit-segment modeling, In *EUROSPEECH 91. 2nd European Conference on Speech Communication and Technology Proceedings*, Genova, Italy, Instituto Int. Comunicazioni.

Martin G. L., Rashid M., 1992, Recognizing overlapping hand-printed characters by centered-object integrated segmentation and recognition, In Moody J. E., Hanson S. J., Lippmann R. P., ed.s, *Advances in Neural Information Processing Systems*, volume 4, Morgan Kaufmann.

Matan O., Burges C. J. C., LeCun Y., Denker J. S., 1992, Multi-digit recognition using a space displacement neural network, In Moody J. E., Hanson

S. J., Lippmann R. P., ed.s, *Advances in Neural Information Processing Systems*, volume 4, Morgan Kaufmann.

Pavlidis T., 1982, *Algorithms for Graphics and Image Processing*, Computer Science Press, Rockville, MD.

R. Allen Wilkinson *et al.* , ed., 1994, *The Second Census Optical Character Recognition System Conference*, U.S. Department of Commerce, National Institute of Standards.

Renals S., Morgan N., 1992, Connectionist probability estimation in hmm speech recognition, Technical Report TR-92-081, International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, USA.

Rumelhart D. E., Hinton G. E., Williams R. J., 1986, Learning representations by back-propagating errors, *Nature*, 323(9):533–536.

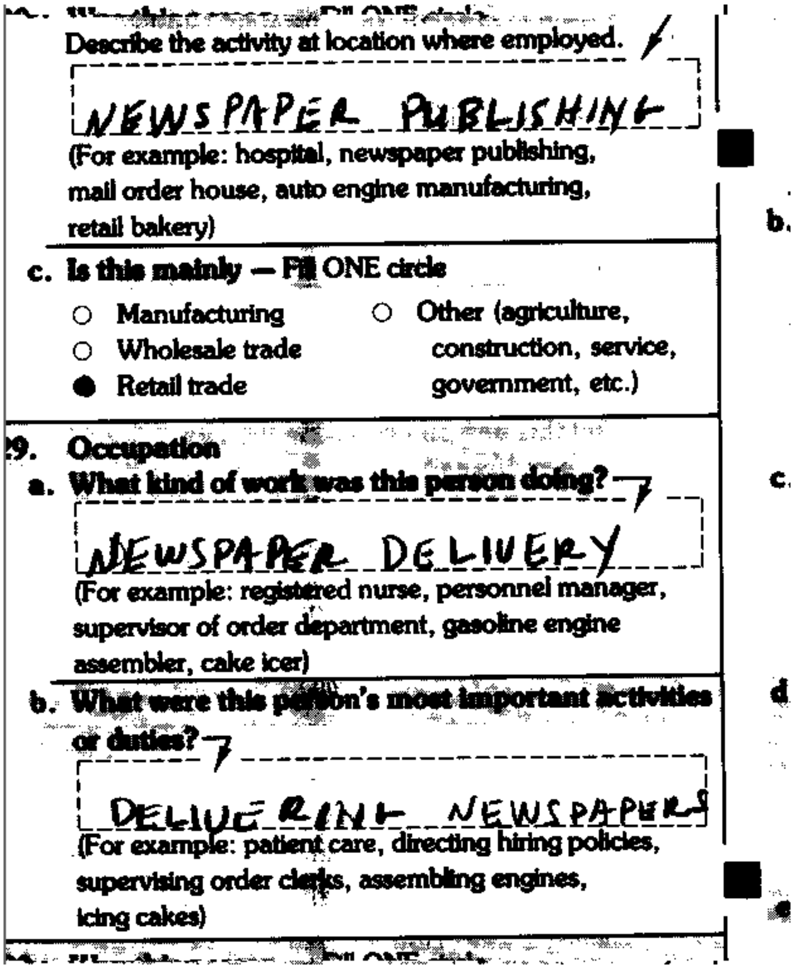
Schenkel M., Weissman H., Guyon I., Nohl C., Henderson D., 1993, Recognition-based segmentation of on-line hand-printed words, In Hanson S. J., Cowan J. D., Giles C. L., ed.s, *Advances in Neural Information Processing Systems*, volume 5, Morgan Kaufmann.

rejected	system	phrase-based	word-based
75%	1.9%	1.9%	5.1%
50%	6.1%	8.9%	19.8%
25%	21.1%	27.0%	36.1%
0%	36.7%	42.0%	50.3%
	(n=1500)		

Table 1: Field error rates under an exact-match criterion after rejecting different fractions of the input. For example, 6.1% error at 50% rejection means that 750 input fields are rejected and 46 out of the remaining 750 accepted input fields are misclassified. The columns "phrase-based" and "word-based" refer to the performance of the subsystems that either only use the phrase-based language model or the word-based language model. The column "system" refers to the performance of the complete system, including decision-tree based integration and rejection.

rejected	system	phrase-based	word-based
75%	1.5%	0.0%	3.7%
50%	3.4%	1.4%	10.6%
25%	12.6%	2.6%	21.0%
0%	29.5%	12.4%	36.5%
	(n=1311/87%)	(n=983/66%)	(n=1311/87%)

Table 2: This table is analogous to Table 1, but recognition rates are only reported for inputs whose transcription is contained in the language model for the given (sub-)system. The last row gives the number of phrases contained in the language model and the corresponding coverage of the language model relative to the complete set of 1500 transcriptions.



Describe the activity at location where employed.

NEWSPAPER PUBLISHING

(For example: hospital, newspaper publishing, mail order house, auto engine manufacturing, retail bakery)

c. Is this mainly — Fill ONE circle

- ○ Manufacturing
- ○ Wholesale trade
- ● Retail trade
- ○ Other (agriculture, construction, service, government, etc.)

9. Occupation

a. What kind of work was this person doing?

NEWSPAPER DELIVERY

(For example: registered nurse, personnel manager, supervisor of order department, gasoline engine assembler, cake icer)

b. What were this person's most important activities or duties?

DELIVERING NEWSPAPERS

(For example: patient care, directing hiring policies, supervising order clerks, assembling engines, icing cakes)

Figure 1: A typical miniform from the database. The input of the system consists of a vertical concatenation of five such miniforms into a large image of about 600×3700 pixels. A significant fraction of miniforms have spurious markings outside the input field, have non-negligible skew, or have been binarized using a threshold that is slightly too low or too high (leading to fading of characters or the presence of noisy areas).

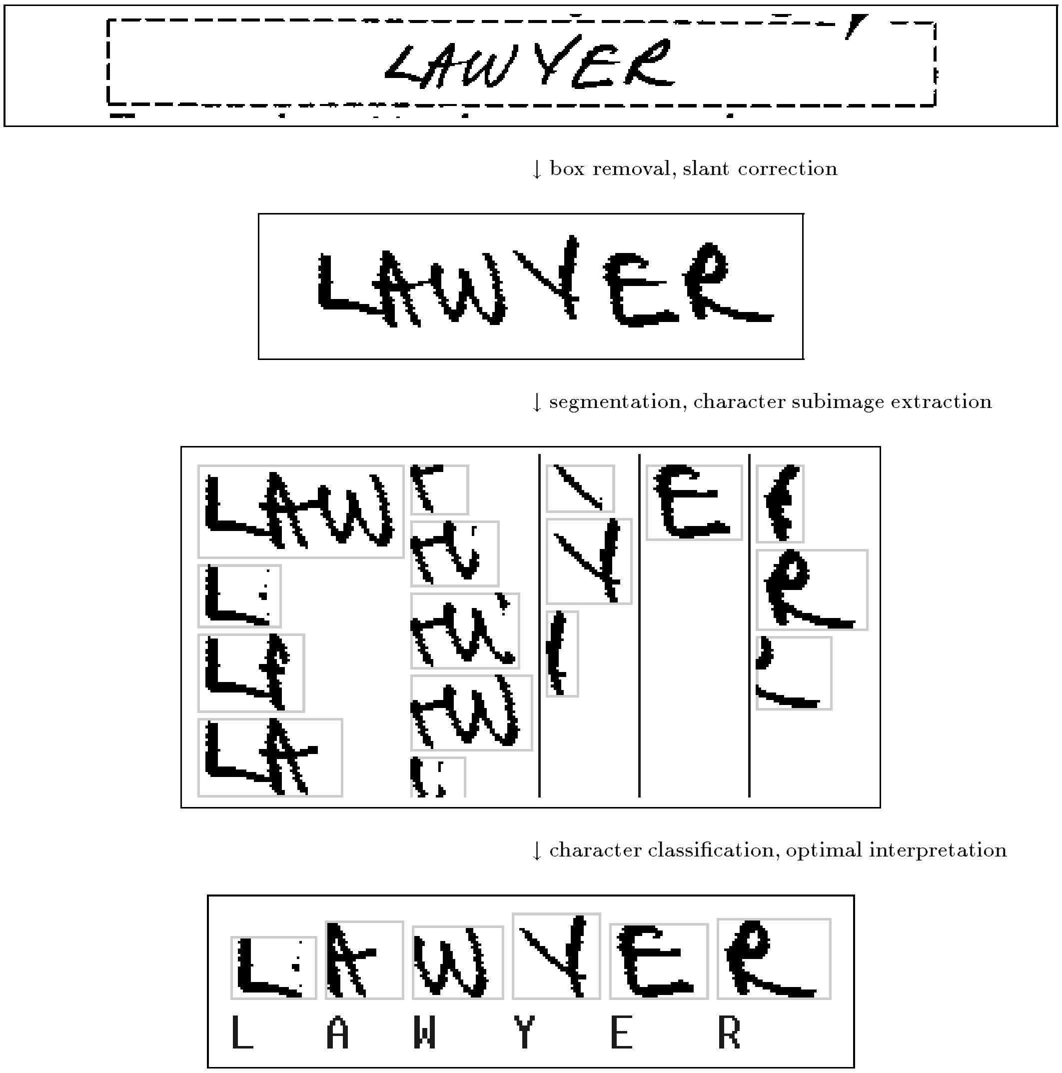


Figure 2: An illustration of the main processing steps of the recognition system. (To keep the figure simple, an unusally easy example was chosen.)